REMARKS

Claims 1-9, 11, 13-23, 28-29, 31-33, 35-36, 38-40, and 42-59 are pending, with claims 1, 7, 11, 15, 19, 28-29, 31-33, 35-36, 38-40, being independent.

Claim 7 is being amended. No new matter has been added. The specification has been amended to correct clerical errors.

Reconsideration of the action and allowance of the claims are requested in light of the foregoing amendments and the following remarks.

Interview Summary

The applicant thanks Examiner Anya for the courtesy of a telephone interview on November 14, 2006. Attending were Examiner Anya and the applicant's representatives, Daniel Burns and Mymy Henderson. The applicant argued that the pending action incorrectly applies the language of independent claim 1 to column 9, lines 1-47 of U.S. Patent No. 5,404,428 to Wu ("Wu") in that the cited portion of the reference does not teach severing dependencies among the dependents of an object when the value of that object changes. The applicant further argued that the pending action also incorrectly applies the language of independent claim 7 to column 8, lines 21-43 and column 11, lines 39-61 of U.S. Patent No. 5,469,538 to Razdow ("Razdow '538") in that the cited portion of the reference does not teach identifying the objects upon which a given object depends as those objects into which the given object passed itself as a requester during execution of a compute method of the given object.

The Examiner stated that claim 7 could be rejected on as not producing a tangible result. In particular, the Examiner suggested reciting maintaining dependencies as a limitation for claim 7.

No agreement was reached as to the sufficiency of any particular claim language or of the allowability of any claim that might be submitted.

Section 103 Rejections

Claims 1, 2, 5-6, 28, 35, 42, and 45-46 stand rejected under 35 U.S.C. §103(a) as allegedly being unpatentable over U.S. Patent No. 5,469,538 to Razdow ("Razdow '538") in view of U.S. Patent No. 5,929,864 to Picott et al. ("Picott") and further in view of U.S. Patent No. 5,404,428 to Wu ("Wu").

Claims 7, 9, 29, 36, and 48 stand rejected under 35 U.S.C. §103(a) as allegedly being unpatentable over Razdow '538 in view of Picott.

Claims 11, 13-18, 31-32, 38-39, 49, and 50-53 stand rejected under U.S.C. §103(a) as allegedly being unpatentable over U.S. Patent No. 6,272,672 B1 to Conway ("Conway") in view of U.S. Patent No. 5,537,593 to Diamond et al. ("Diamond ") and further in view of Wu.

Claims 19-23, 33, 40, and 54-59 stand rejected under 35 U.S.C. §103(a) as allegedly being unpatentable over Conway in view of U.S. Patent No. 5,526,475 to Razdow ("Razdow '475") and further in view of U.S. Patent No. 5,689,711 to Bardasz et al. ("Bardasz").

Claims 3, 4, and 43-44 stand rejected under 35 U.S.C. §103(a) as allegedly being unpatentable over Razdow '538 in view of Picott and further in view of Wu as applied to claim 1, and further in view of Conway.

Claims 8 and 47 stand rejected under 35 U.S.C. Section 103(a) as allegedly being unpatentable over Razdow '538 in view of Picott as applied to claim 7, and further in view of Conway.

Claims 1, 2, 5-6, 28, 35, 42, and 45-46 stand rejected in view of Razdow '538, Picott, and Wu. Claim 1 recites in part, "when the value of object B changes, invalidating the dependents of object B and all of their further dependents, including severing dependencies among the dependents of object B and all of their further dependents." The relied upon portions of the cited references alone and in combination fail to teach or suggest this feature of claim 1.

The Examiner again conceded that Razdow '538 is silent with respect to severing dependencies. *See* Office Action mailed 9/22/06, pp. 2-3, ¶4. The cited portion of Picott merely describes message sending between nodes in a directed acyclic graph (Picott, col. 7, lines 28-35):

> When DAG node 230 requests redraw of dependency node A, dependency node A sends an "evaluate me" message to dependency node B. Dependency node B then evaluates and returns results to dependency node A over communication channel 220. Note that dependency node A knows how to control DAG node 230. Thus, when data is returned from dependency node B, dependency node A can pass the data onto DAG node 230.

The relied upon portion of Wu teaches that when a view model attribute is modified by an application program, derived items dependent upon that attribute are invalidated but not severed from the acyclic graph. *See* Wu, col. 9, lines 1-47. The cited portion of Wu discloses maintaining dependencies in the acyclic graph, not severing dependencies. This is clear from the relied upon portion of Wu, which states that invalid derived items are calculated "by descending the acyclic graph until valid item(s) are reached, and then calculating invalid item(s) back up the acyclic graph." Wu, col. 9, lines 8-39. If Wu taught severing dependencies, invalid derived items could not be reached by traversing the acyclic graph, because the links would be severed.

With respect to the cited portion of Wu, the Examiner contended in the Response to Arguments that (Office Action mailed 9/22/06, p. 15, ¶48):

> the invalidation of dependent derived items includes severing the dependent derived items because Merriam Webster's Collegiate Dictionary (10th edition) describes invalidate as follows: to weaken or destroy the cogency . . . NULLIFY and the Examiner believes that destroying/nullifying dependent derived items implies severing the dependent derived items.

The applicant respectfully disagrees. Regardless of how "invalidate" is defined in the dictionary, the cited portion of Wu does not disclose severing dependencies; Wu teaches maintaining dependencies. Moreover, the relied upon portion of Wu could not be read to disclose severing dependencies, because invalidated derived items are calculated by traversing the acyclic graph with dependencies intact.

Accordingly, claim 1 and its dependents are in condition for allowance. Claims 28, 35, 42, and 45-46 include limitations analogous to those of claim 1 and are in condition for allowance for at least the same reasons.

Claims 7, 9, 29, 36, and 48 stand rejected in view of Razdow '538 and Picott. Claim 7 recites in part, "identifying the objects upon which a given object depends as those objects into which the given object passed itself as a requester during execution of a compute method of the

given object." The relied upon portions of the cited references alone and in combination fail to teach or suggest this feature of claim 7.

The cited portions of Razdow '538 describe representing numeric equations as dependency graphs, but do not discuss this feature, much less object oriented programming. *See* Razdow '538, col. 8, lines 21-43; col. 11, lines 39-61. Likewise, the relied upon portion of Picott merely describes message sending between nodes in a directed acyclic graph. *See* Picott, col. 7, lines 28-35.

In the Response to Arguments, the Examiner asserted that "for the expression (x+l).y = 24 of the column 11 lines 39 - 61 of the Razdow prior art reference, x and y are requestors object. This is because x and y respectively request for the values 5 and 4 that are used to evaluate/compute the expression." Office Action mailed 9/22/06, p. 16, ¶48. The applicant respectfully disagrees. The relied upon portion of Razdow '538 does not disclose x passing itself as a requester during execution of a compute method into an object upon which x depends, nor is this disclosed for y. For example, the expression y:=x-1 of Razdow '538 would require y to pass itself as a requester into x, upon which y depends, during execution of a compute method. However, the cited portion of Razdow '538 merely states that "the nodes of the SDG [Symbolic Dependency Graph] 26 represent expressions qua expressions and the arcs represent variable names that represent the corresponding expressions." Therefore, the relied upon portion of Razdow '538 does not teach or suggest this feature of claim 7.

In the interview, the Examiner stated that claim 7 could be rejected on as not producing a tangible result. In particular, the Examiner suggested reciting maintaining dependencies as a limitation for claim 7. The applicant respectfully demurs to the notion that identifying objects and marking other objects as dirty does not create a tangible result. Nevertheless, to expedite prosecution, claim 7 has been amended and now recites modifying a dependency graph by severing dependencies among dependents of the identified objects with changed value and all of their further dependents to create a modified dependency graph.

Accordingly, claim 7 and its dependents are in condition for allowance. Claims 29, 36, and 48 include limitations analogous to those of claim 7 and are in condition for allowance for at least the same reasons.

Claims 11, 13-18, 31-32, 38-39, 49, and 50-53 stand rejected in view of Conway, Diamond and Wu. Claim 11 recites in part, "receiving a change to a value of a changed object, the changed object having objects depending directly on the changed object and objects depending indirectly on the changed object through an object different from the changed object... severing dependencies from the changed object and all of its direct and indirect dependent objects."

The Examiner again conceded that Conway is silent with respect to severing dependencies. *See* Office Action mailed 9/22/06, p. 7, ¶17. But the Examiner argued that Diamond remedies the deficiency in Conway. The relied upon portion of Diamond is as follows (Diamond, col. 8, lines 45-49, *emphasis added*):

> The "re-evaluate bounds downward" message is used to propagate downward flowing bounds principally in heterogeneous graphs or trees, e.g., mini-max decision processes. This type of graph or tree is set by the parameters entered by the user. D-bounds are used to promote pruning of nodes located on paths other than that of the originating node. These pruning actions take place when these more or less global bounds are passed downwardly from parent to offspring where a determination is made to sever the link between offspring and parent.

Diamond discloses that pruning is triggered in response to a "re-evaluate bounds upward" message generated by a node when the bounds of the node have changed. *See* Diamond, col. 8, lines 33-35 and 50-56. The relied upon portion of Diamond does not disclose severing dependencies from the changed object. In contrast, Diamond discloses that node pruning removes nodes located on paths other than that of the originating node: "[t]he end result is that bounds originating on one path are propagated down another path, resulting in a cut off or pruning action." Diamond, col. 9, lines 6-9.

In the Response to Arguments, the Examiner asserted that (Office Action mailed 9/22/06, p. 16, ¶48):

> pruning is triggered in response to a "re-evaluate upward". The "re-evaluate upward" is message that indicate a change in the offspring node/object. In effect the pruning/severing is triggered in response to an indication of a change in the offspring node/object, thus covering the invention as claimed.

The applicant respectfully disagrees. Diamond teaches that the "originating node" is the offspring node with a changed bound. However, the "originating node" is on a different path

than the node which later is pruned. *See* Diamond, col. 9, lines 6-9. Because the pruned nodes are on different paths than the originating node, these pruned nodes do not depend, either directly or indirectly, on the originating node. Therefore, because the pruned nodes are independent of the originating node with the changed bound, the relied upon portions of Diamond do not teach severing dependencies from the changed object, as recited in claim 11.

As addressed above, the relied upon portion of Wu teaches that when a view model attribute is modified by an application program, derived items dependent upon that attribute are invalidated but not severed from the acyclic graph. *See* Wu, col. 9, lines 1-47.

Accordingly, neither Conway, Diamond, nor Wu, alone or in combination, render claim 11 obvious. For at least these reasons, claim 11 and its dependents are in condition for allowance. Claims 15-18, 31-32, 38-39, 49, and 50-53 include analogous limitations and are therefore in condition for allowance for at least the same reasons as claim 11.

Claims 19-23, 33, 40, and 54-59 stand rejected in view of Conway, Razdow '475, and Bardasz. Claim 19 recites in part, "calculating the dependency among objects in the set of objects dynamically at the time objects calculate their values."

Conway fails to teach or suggest this feature, as does Bardasz. The cited portion of Conway merely describes message sending between components during recomputation of outputs. *See* Conway, FIG. 6, and col. 21, lines 1-46. According to Bardasz, "[d]ata objects are components in the graph that contain values." Bardasz, col. 6, lines 46-47. "Operators are components in the graph that can take action on at least one data object and are functions or evaluable expressions." Bardasz, col. 6, lines 50-53. The relied upon text in Bardasz discloses determining dependencies of operators on data objects. *See* Bardasz, col. 20, lines 18-25. However, the dependencies between operators and data objects already exist in the dependency graph of Bardasz *before* operator components are evaluated.

The relied upon portion of Razdow '475 reads as follows (Razdow '475, col. 4, line 63 – col. 5, line 10, *emphasis added*):

> The expression compiler 14 generates a new value of the modified expression. Next, the expression compiler 14 automatically goes down the linked list that represents the numerical dependency graph and marks as "out of date" all expressions that depend on the modified expression. Next, each node that has been marked out of date in turn recalculates itself (with the assistance of the

> numerical computational engine 18). In performing each recalculation, the node looks up the graph 16 to find the new values of an recalculated nodes (i.e., expressions). When a node is recalculated, the editor automatically updates the document 12. Accordingly, the document 12 is always the most current representation of the nodes of the numerical dependency graph 16. The numerical computational engine 18 has access to a numerical library 20 that is a library of subroutines for performing numerical computations.

The above quoted passage from Razdow '475 fails to teach or suggest calculating the dependency among objects in the set of objects *dynamically* at the time objects calculate their values. As was the case with Bardasz, the dependencies already exist in the dependency graph *before* the expression is evaluated. This is apparent since the expression compiler 14 traverses "a linked list that represents the numerical dependency graph."

In the Response to Arguments, the Examiner asserted that "[t]he Razdow prior art teaches this limitation because the dependency graph is calculated automatically when a **new value** is generated." Office Action mailed 9/22/06, p. 16, ¶48 (emphasis in original). The applicant respectfully disagrees. Although Razdow '475 discloses generating a new value automatically, the *dependency* among the objects is not calculated dynamically, because the pre-existing linked list representing the numerical dependency graph is traversed in generating the new value. In other words, the dependency among objects already exists and is not calculated dynamically at the time objects calculate their values, as recited in claim 19.

Accordingly, Conway, Razdow '475, and Bardasz, alone or in combination, do not render claim 19 obvious. For at least these reasons, claim 19 and its dependents are in condition for allowance. Claims 33, 40, and 54-57 include analogous limitations and are therefore in condition for allowance for at least the same reasons as claim 19.

Claims 3, 4, and 43-44 stand rejected in view of Razdow '538, Picott, Wu, and Conway. Claim 3 depends from claim 1. As addressed above, Razdow '538, Picott, and Wu fail to teach or suggest "when the value of object B changes, invalidating the dependents of object B and all of their further dependents, including severing dependencies among the dependents of object B and all of their further dependents," as required by claim 1. The relied upon portion of Conway fails to remedy this deficiency.

Accordingly, claim 3 is in condition for allowance. Claims 4 and 43-44 include analogous limitations and are in condition for allowance for at least the same reasons.

Claims 8 and 47 stand rejected in view of Razdow '538, Picott, and Conway. Claim 8 depends from claim 7. As addressed above, Razdow '538 and Picott fail to teach or suggest "identifying the objects upon which a given object depends as those objects into which the given object passed itself as a requester during execution of a compute method of the given object," as required by claim 7. The relied upon portion of Conway fails to remedy this deficiency.

Accordingly, claim 8 is in condition for allowance. For at least the same reasons set forth above with respect to claim 7, claim 47 is also in condition for allowance.

Conclusion

For the foregoing reasons, the applicant submits that all the claims are in condition for allowance.

By responding in the foregoing remarks only to particular positions taken by the examiner, the applicant does not acquiesce with other positions that have not been explicitly addressed. In addition, the applicant's arguments for the patentability of a claim should not be understood as implying that no other reasons for the patentability of that claim exist.

Please apply any charges or credits to deposit account 06-1050.

Respectfully submitted,

Date: 22 November 2006

Daniel J. Burns
Reg. No. 50,222

Customer No.: 021876
Telephone: (650) 839-5070
Facsimile: (650) 839-5071